Sequence Mining Without Sequences: a New Way for Privacy Preserving

Stéphanie Jacquemont, François Jacquenet, Marc Sebban
Université Jean Monnet de Saint-Etienne
EURISE, 23 rue du docteur Paul Michelon
42023 Saint-Etienne Cedex 2
{stepjacq,jacquene,sebbanma}@univ-st-etienne.fr

Abstract

During the last decade, sequential pattern mining has been the core of numerous researches. It is now possible to efficiently discover users' behavior in various domains such as purchases in supermarkets, Web site visits, etc. Nevertheless, classical algorithms do not respect individual's privacy, exploiting personal information (name, IP address, etc.). We provide an original solution to privacy preserving by using a probabilistic automaton instead of the original data. An application in car flow modeling is presented, showing the ability of our algorithm to discover frequent routes without any individual information. A comparison with SPAM *is done showing that even if we sample from the automaton, our approach is more efficient.*

1 Introduction

Sequential pattern mining aims at automatically finding subsequences (or patterns) that appear *frequently* (*i.e.* more than a support threshold) in a database of sequences. One of the algorithmic difficulties is that each pattern can be made up of non-consecutive elements in the original sequence. Many algorithms have been proposed during the last decade, such as [14, 20, 8, 16]. There have also been a lot of applications based on such techniques [9]. One of the first concerned the discovery of customer's behavior in supermarkets [1] in the form of subsequences of purchases. That knowledge may be very valuable for marketing department of the companies. In the domain of manufacturing supervision, sequence mining can be used to discover frequent patterns of alarms, what may help supervisors while searching for defaults in the plant [12]. More recently, in the domain of Web Mining [13], various systems have been designed for making use of Web site logs in order to model the behavior of Web users [17]. Another application, for which we implemented a prototype in this paper, is the discovery of routes frequently used by car drivers in towns.

These numerous applications prove that sequence mining algorithms are useful tools for discovering knowledge in various situations. Nevertheless, in the case we try to discover knowledge from the observation of human behavior, we claim that these algorithms do not preserve individual's privacy. For example, regarding the problem of discovering frequent sequences of Web pages visited throughout time, the input data are, at least, the IP address of the visitors and the Web pages they have browsed. This is a great breach to their privacy and users might want their private information not to be logged on the servers. In the case of frequent routes in towns, to get interesting knowledge, one could install many Web cams in the town and trace the route for each driver. This is obviously a non acceptable breach to the privacy of car drivers.

Privacy preserving data mining is a new subject of research that appeared to be essential for some years. There is a great demand from users, and more generally from the society, that data miners preserve their privacy. As stated by [19], *the main objective in privacy preserving data mining is to develop algorithms for modifying the original data in some way, so that the private data and private knowledge remain private even after the mining process.* A huge number of papers have been published during the last five years, for example [2, 18, 7, 4]. Considering the discovery of frequent routes traversed by car drivers, to preserve their privacy, we may think about de-identifying images from Web cams such as in [15], but this is a costly solution. Moreover, installing a Web cam in every street of a town may lead to psychological resistance from people.

In this paper, we propose a costless solution to privacy preserving for problems that may be stated as flow control problems, that is the case of frequent path discovery in Web sites and frequent route discovery in towns. We propose to model this flow of data in the form of a weighted automaton

for which we provide a probabilistic solution to discover frequent patterns (potentially with gaps) under constraints, without any information about the original data.

In Web usage mining, the states of this automaton are the pages of the site and transitions the hyperlinks between pages. The weight assigned to each transition corresponds to the number of users who clicked on the corresponding hyperlink and the weight of each state is the number of times users left the web site from the corresponding page. Therefore, for the users, we do not need anymore their IP address and the Web pages they visited, but only counters on each transition and state. [5] exploited this idea defining the concept of *composite association rule* processed from a structured directed graph built from the log files of the Web site. More recently, [6] show how to use higher-level Markov models in order to process a weighted automaton to discover frequent paths of Web site users from log files. Nevertheless, these works are more restricted than our approach in the sense that first, they need the Web log files and second, they aim at discovering sequential patterns made up of consecutive Web pages while we are able to discover non consecutive ones.

The problem of discovering frequent routes in a town can also be represented in the form of an automaton. Considering a map of a part of a town, the non-initial and non-final states model the crossroads. The initial and final states respectively represent the entry and exit gates in the map. The transitions model the streets. The weights are obtained by using counters on transitions and on the final states (the other states having a null counter). For each car, we do not need anymore its license and the streets it traverses. We will show that it is possible to discover frequent routes just using the automaton. It is essential to be able to discover frequent non consecutive patterns for example to set up some combined advertisements campaign, for improving traffic, etc.

The rest of this paper is organized as follows. Section 2 defines the way we replace the database of sequences by a probabilistic automaton. This idea was originally proposed in [10] and exploited in [11]. We extend it, in Section 3, by using such an automaton-based structure to propose a *constrained* sequence mining algorithm in the context of *privacy preserving*. Section 4 describes a traffic simulator able to discover frequent routes in towns. Beside the description of the system, we experimentally show that our algorithm is more efficient than sampling sequences from the automaton and then using a classical sequence mining algorithm.

2 Using a PDFA instead of Sequences

2.1 Definitions and Notations

Achieving a sequence mining task without the original sequences requires the use of a suited structure able to express the same knowledge as the one underlying in the sequences themselves. We propose here to use a probabilistic deterministic finite state automaton (PDFA).

Definition 1 *a* PDFA $A =< Q, \Sigma, q, q_0, \pi, \pi_F >$ *is a tuple where* Q *is a finite set of states;* Σ *is the alphabet;* $q : Q \times \Sigma \rightarrow Q$ *is a transition function;* q_0 *is the initial state;* $\pi : Q \times \Sigma \rightarrow [0, 1]$ *is a transition function;* $\pi_F : Q \rightarrow [0, 1]$ *is a probability function giving to each state a probability to be final.*

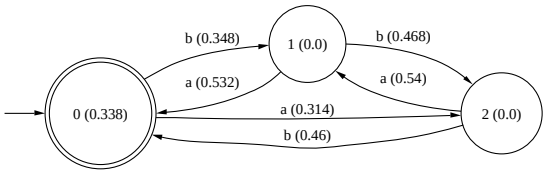

Figure 1. An example of PDFA.

Fig.1 shows a PDFA where $Q = \{0, 1, 2\}$, $\Sigma = \{a, b\}$, $q_0 = 0$, and for instance $q(0, a) = 2$, $\pi(0, a) = 0.314$ and $\pi_F(0) = 0.338$. In this paper, we assume that we have a weighted automaton of the problem we tackle. We showed in introduction that the "real life" can provide such automata, that is the case of a road traffic. To be a PDFA, we must normalize the counters of the automaton to obtain a probability distribution as follows:

$$\pi(S, z) = \frac{n(S, z)}{\sum_{z' \in \Sigma \cup \{\#\}} n(S, z')}, \forall z \in \Sigma \cup \{\#\}, \forall S \in Q \quad (1)$$

where $n(S, z)$ is the counter of the transition $q(S, z)$, $\#$ is a termination symbol, and $n(S, \#)$ is the counter of state S (note that $\pi(S, \#) = \pi_F(S)$).

2.2 Estimation of Pattern Probabilities

If we aim at using a PDFA for sequence mining, it must first allow us to correctly estimate the true probability (*i.e.* in the hidden original sequences) of any pattern. Let us show now that using formulas proposed by Hingston in [10], we can correctly assess the true probability of any pattern. Then, we will generalize them to build an efficient sequence mining algorithm that will allow us not only to do without the original data but also to impose constraints on the extracted patterns.

Let $A =< Q, \Sigma, q, q_0, \pi, \pi_F >$ be a PDFA. To estimate (with $\hat{p}(x)$) the unknown (or hidden) proportion $p(x)$ of sequences that contain a letter x, Hingston defines the probability $P(S, x)$ that a path in A starting from state S contains an x. This is ensured either if a path begins with an x (of

probability $\pi(S,x)$), or with some other symbol $z \in \Sigma$ and is followed by a path starting at the next state (given by $q(S,z)$) and containing an x. This can be written with the recursive formula:

$$P(S,x) = \pi(S,x) + \sum_{z \neq x \in \Sigma} (\pi(S,z) \times P(q(S,z),x)) \quad (2)$$

that one can rewrite as follows:

$$P(S,x) = \pi(S,x) + \sum_{T \in Q} \left(\sum_{z \neq x, q(S,z)=T} \pi(S,z) \right) \times P(T,x). \quad (3)$$

If $S = q_0$, $P(S,x)$ represents $\hat{p}(x)$. Computing $P(S,x)$ requires to handle a system of linear equations that can be efficiently solved with matrix products. Let $\rho(x)$ be the matrix of components $\rho_{S,T}(x) = \sum_{z \neq x, q(S,z)=T} \pi(S,z)$ describing the probability to use a transition different from x between states S and T. Let $P(x)$ (resp. $\pi(x)$) be the vector of values of $P(S,x)$ (resp. $\pi(S,x)$), Eq.3 becomes:

$$P(x) = \pi(x) + \rho(x)P(x) = (I - \rho(x))^{-1}\pi(x), \quad (4)$$

where I is the identity matrix. Let us take an example with PDFA of Fig.1 and estimate with $\hat{p}(a)$ the unknown probability $p(a)$ of sequences that contain the letter a. Vector $\pi(a)$ has the components $\pi(0,a) = 0.314$, $\pi(1,a) = 0.532$, $\pi(2,a) = 0.54$. For matrices $\rho(a)$ and $(I - \rho(a))^{-1}$, we get

$$\rho(a) = \begin{pmatrix} 0 & 0.348 & 0 \\ 0 & 0 & 0.468 \\ 0.46 & 0 & 0 \end{pmatrix}$$

$$and \quad (I - \rho(a))^{-1} = \begin{pmatrix} 1.081 & 0.376 & 0.176 \\ 0.233 & 1.081 & 0.506 \\ 0.498 & 0.173 & 1.081 \end{pmatrix}.$$

We deduce that $\hat{p}(a) = P(0,a) = 0.635$. Based on the same principle, one can estimate the proportion of sequences that contain a pattern $w =< x_1 ... x_l >$ composed of l symbols *potentially non consecutive*. Let $F(S,T,x_1)$ be the probability that a random path starting at state S and ending at state T contains exactly one symbol x_1. Hingston uses similar reasoning as in $P(x)$ to show that:

$$F(S,T,x_1) = \sum_{x_j \neq x_1} (p(S,x_j) \times F(q(S,x_j),T,x_1)) + p(S,x_1), \quad if \quad q(S,x_1) = T \quad and \quad 0 \quad otherwise. \quad (5)$$

One can rearrange Eq.5 using a matrix $\gamma(x_1)$ of values $\gamma(S,T,x_1) = p(S,x_1) \ \ if \ q(S,x_1) = T \ and \ 0 \ otherwise.$ Writing $F(x_1)$ for the matrix of values $F(S,T,x_1)$, Eq.5 becomes: $F(x_1) = \gamma(x_1) + \rho(x_1)F(x_1)$ and as before, we deduce $F(x_1) = (I - \rho(x_1))^{-1}\gamma(x_1)$.

Returning to our objective, we aim at computing $P(S, < x_1 ... x_l >)$, the probability of any pattern starting at state S. Let us focus on the case $l = 2$, *i.e.* $P(S, < x_1 x_2 >)$. Note that a sequence containing an x_1 followed later by an x_2 can be divided into one part containing the first x_1 in the sequence, and the following part, which contains an x_2. We can deduce that:

$$P(S, < x_1 x_2 >) = \sum_T F(S,T,x_1)P(T,x_2). \quad (6)$$

Using a matrix form, $P(< x_1 x_2 >) = F(x_1)P(x_2)$. Generalizing, we get

$$P(S, < x_1 ... x_l >) = F(x_1) \times ... \times F(x_{l-1}) \times P(x_l). \quad (7)$$

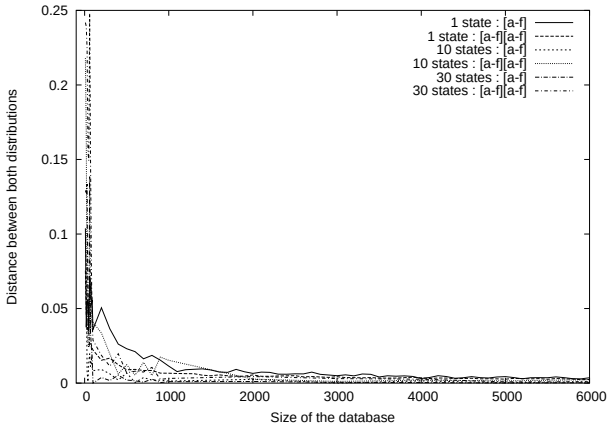

Figure 2. Average difference between $\hat{p}(< x_1 x_2 >)$ and $p(< x_1 x_2 >)$ according to regular expressions on Σ.

To assess the efficiency of a PDFA to estimate the true probabilities of the hidden sequences, we implemented these formulas and carried out a series of experiments. We simulated several target distributions, from an alphabet $\Sigma = \{a,b,c,d,e,f\}$, in the form of automata with 1, 10 or 30 states. From each automaton, we sampled sets of sequences of different sizes (from 10 to 6,000), and for each of them, we computed $\hat{p}(< x_1 x_2 >) \ \forall x_1, x_2 \in \Sigma \cup \{\lambda\}$ (where λ is the empty symbol), and compared it with the true probability $p(< x_1 x_2 >)$ observed in the set of sequences. Fig.2 shows the average difference between the estimated and the true probabilities. We note that in all cases it converges rapidly toward 0. In other words, if the PDFA models a sufficiently large number of hidden sequences, we can claim that it correctly estimates the probability of any pattern, and so we can build a PDFA-based sequence mining algorithm.

3 Completeness and Correctness

In a PDFA-based sequence mining algorithm, the test deciding if a pattern $w =< x_1...x_l >$ is frequent will not be done using its unknown probability $p(w)$ but its estimate $P(q_0, < x_1...x_l >)$. How can we ensure that the decision taken is right or wrong? We provide here a lower bound of the number of sequences, on which the PDFA must be built, to ensure a given level of completeness and correctness. Let us design the following statistical test. Accepting a pattern w as being frequent means that the unknown probability $p(w)$ must be at least equal to a fixed support threshold σ_0. Since $p(w)$ is unknown, one must formulate a null hypothesis H_0 on its value. Here, we fix $H_0 : p(w) = \sigma_0$. Writing H_0 like that rather than $H_0 : p(w) > \sigma_0$ is necessary because the establishment of a null hypothesis requires to fix a specific value on the theoretical parameter. However, we are only interested here in rejecting H_0 (in favor of an alternative hypothesis H_a) in one direction, *i.e.* the case where $p(w)$ is in fact smaller than σ_0. So, the main feature of the test is its alternative hypothesis $H_a : p(w) < \sigma_0$.

When a statistical test is carried out, there are two kinds of possible errors: First, a true null hypothesis can be incorrectly rejected and second, a false null hypothesis can fail to be rejected. The former error is called a Type I error (usually designated by α) and the latter error is called a Type II error (usually called β). α corresponds to the risk to *reject a true frequent pattern*, that statistically defines the non completeness of the algorithm. More formally, $\alpha = P(\hat{p}(w) < k | H_0\ true)$. When the number of sequences is higher than 30, it is known that a proportion follows a normal law such that: $\hat{p}(w) \approx N(p(w), \sqrt{\frac{p(w)(1-p(w))}{N}})$ (N will be estimated by $\hat{N}$ in the following of the paper). α can be rewritten as

$$\alpha = P(\frac{\hat{p}(w) - \sigma_0}{\sqrt{\frac{\sigma_0(1-\sigma_0)}{N}}} < \frac{k - \sigma_0}{\sqrt{\frac{\sigma_0(1-\sigma_0)}{N}}}). \quad (8)$$

We can easily deduce the bound k which corresponds to the $(1-\alpha)$-percentile z_α of the normal law.

$$k = \sigma_0 - z_\alpha \sqrt{\frac{\sigma_0(1-\sigma_0)}{N}}. \quad (9)$$

As defined above, β corresponds to the risk to *accept a false frequent pattern*, that statistically defines the non correctness of the algorithm. Since $H_a : p(w) < \sigma_0$ is true here, we have to fix a given value for $p(w)$ satisfying the constraint $p(w) < \sigma_0$. Let $\sigma_a < \sigma_0$ be this value. More formally, $\beta = P(\hat{p}(w) > k | H_a\ true)$. As previously done for α,

$$\beta = P(\frac{\hat{p}(w) - \sigma_a}{\sqrt{\frac{\sigma_a(1-\sigma_a)}{N}}} > \frac{k - \sigma_a}{\sqrt{\frac{\sigma_a(1-\sigma_a)}{N}}}). \quad (10)$$

We can finally deduce that

$$k = \sigma_a - z_\beta \sqrt{\frac{\sigma_a(1-\sigma_a)}{N}}. \quad (11)$$

Theorem 1 *To ensure a completeness of (100-α)% and a correctness of (100-β)% according to a given support threshold σ_0, the lower bound on the number of sequences N_{LB} on which the* PDFA *must be built is equal, $\forall \sigma_a \in [0,1] | \sigma_a < \sigma_0$, to*

$$N_{LB} = [\frac{z_\beta \sqrt{\sigma_a(1-\sigma_a)} + z_\alpha \sqrt{\sigma_0(1-\sigma_0)}}{\sigma_0 - \sigma_a}]^2,$$

Proof 1 *The proof is straightforward using Eq.9 and 11. Actually, we can deduce that*

$$\sigma_0 - z_\alpha \sqrt{\frac{\sigma_0(1-\sigma_0)}{N}} = \sigma_a - z_\beta \sqrt{\frac{\sigma_a(1-\sigma_a)}{N}}.$$

Extracting N from this equation, we obtain the lower bound.

4 Constraint-based Sequence Mining

The use of constraints is one of the current trend in sequence mining. Length and width restrictions, minimum or maximum gap between elements, time window of occurrence, or regular expressions [20, 8, 16] are used to reduce the number of frequent patterns. Since we do not have the original sequences, we propose to use the information provided by the PDFA to constrain the extracted patterns.

4.1 Relevance of the Patterns

The aim is here to use the values $P(q_0, < x_1...x_l >)$ to assess the statistical relevance of a frequent pattern. Roughly speaking, we mean that a frequent pattern does not always express a significant information. Since tuning the support threshold is a tricky task, we propose to constrain, using two statistical tests, a sequence to be not only frequent but also statistically relevant.

Proportion Constraint. The first test verifies an *absolute* condition: a pattern $w =< x_1...x_l >$ must cover a significant part of the probability density of all hidden sequences. To ensure this constraint, we apply a proportion test (called PROP_TEST) aiming at verifying if $P(q_0, < x_1...x_l >)$ (that we will call $\hat{p}(w)$) is high enough. To do this, we test the null hypothesis $H_0 : p(w) = 0$, against the alternative one $H_a : p(w) > 0$ (where $p(w)$ is the unknown probability). If the number of sequences N is large enough[1], $\hat{p}(w)$

[1] Since we don't use the sequences, N is unknown. We will explain how to estimate N using the PDFA.

asymptotically follows the normal law. Let us determine the threshold k which defines the bound of rejection of H_0, and which corresponds to the $(1-\alpha)$-percentile U_α of the distribution of $p(w)$ under H_0. We can show that $P(\hat{p}(w) > k) = \alpha$ iff $k = U_\alpha \sqrt{\frac{\hat{p}(w)(1-\hat{p}(w))}{N}}$. We get the decision rule: *if $\hat{p}(w) > k$, the proportion constraint on w is satisfied.*

Estimation of N. To assess the unknown value N, we are using the PDFA and computing an estimate $\hat{N}$. Let μ be the unknown average size of the hidden sequences. The estimate of N is $\hat{N} = \frac{n}{\hat{\mu}}$, where n is the number of letters used in the sequences and $\hat{\mu}$ is the estimate of μ. In fact, n is computable with the formulae $n = \sum_{S \in Q} \sum_{z \in \Sigma} n(S, z)$, using $n(S, z)$ as defined in Section 2.1. It remains to compute $\hat{\mu}$, such that:

$$\hat{\mu} = \sum_{\delta=0}^{\infty} \delta \times P(q_0, size = \delta) \quad (12)$$

where $P(q_0, size = \delta)$ is the probability of a hidden sequence to have δ letters. Let $\tau_{S,T} = \sum_{z, q(S,z)=T} \pi(S, z)$ be the probability to use one transition between states S and T. We can establish that:

$$P(S, size = \delta) = \sum_{T \in Q} \tau_{S,T} \times P(T, size = \delta - 1). \quad (13)$$

This is a geometric series of common ratio τ and first term $P(S, size = 0) = \pi_F(S)$. Using $P(q_0, size = \delta)$ and Eq.12 we deduce that:

$$\hat{N} = \frac{\sum_{S \in Q} \sum_{z \in \Sigma} n(S, z)}{\sum_{\delta=0}^{\infty} \delta \times P(q_0, size = \delta)}. \quad (14)$$

We carried out experiments to show that Eq.14 provides a correct estimate of N. We sampled several sets of sequences (of size N from 100 to 6,000) from the PDFA of Fig.1. For each set, we compared N and $\hat{N}$. Fig.3 shows that $\hat{N}$ provides a good estimation of N, and then can be efficiently used in PROP_TEST.

Dependence Constraint. We also propose here to verify a relative condition, *i.e.* if there exists a statistical dependence between w and the pattern $w' =< x_1 ... x_{l-1} >$. Roughly speaking, the majority of the sequences that contain w' must also satisfy $w =< w' > . < x_l >$, where "." is the concatenation function. This dependence can be assessed by analyzing the nature of x_l occurring after w'. We generate an output vector $\overrightarrow{V_{out}}$ of dimension $|\Sigma \cup \{\#\}|$. Each component $\overrightarrow{V_{out}}(i)$ is the expected number of sequences that have the symbol $z_i \in \Sigma \cup \{\#\}$ which follows the pattern w'. It means that $\overrightarrow{V_{out}}(i) = P(q_0, < x_1 ... x_{l-1} z_i >) \times \hat{N}$. We arrange $\overrightarrow{V_{out}}$ such that the considered symbol x_l is the first component of the vector. Then we test the dependence between $\overrightarrow{V_{out}}$ and an input vector $\overrightarrow{V_{in}}$ for which the first component is the expected number of sequences that contain the pattern w' (the other components are null). To do this, we run a Chi-square test (called CHI2_TEST) from $\overrightarrow{V_{in}}$ and $\overrightarrow{V_{out}}$, building the following statistic χ^2:

$$\chi^2 = \sum_{i=1}^{|\Sigma \cup \{\#\}|} \frac{[(\overrightarrow{V_{in}}(i) - \Psi(i))^2 + (\overrightarrow{V_{out}}(i) - \Psi(i))^2]}{\Psi(i)},$$

where $\Psi(i) = \frac{\overrightarrow{V_{in}}(i) + \overrightarrow{V_{out}}(i)}{2}$. χ^2 follows a Chi-square distribution with $2 \times |\Sigma \cup \{\#\}| - 1$ degrees of freedom. It is then possible to test if χ^2 is smaller than $X_\alpha^{2 \times |\Sigma \cup \{\#\}| - 1}$, which is the $(1-\alpha)$-percentile of the Chi-square law. We get the decision rule: *if $\chi^2 < X_\alpha^{2 \times |\Sigma \cup \{\#\}| - 1}$, the dependence constraint is verified.* Combining the two constraints, we can define a frequent and relevant pattern.

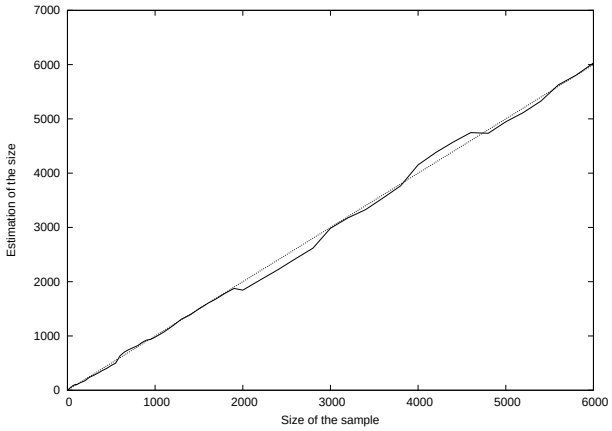

Figure 3. Comparison between N and $\hat{N}$. The dotted line is the situation where $\hat{N}$ would perfectly estimate N. The solid line is obtained with the series of experiments.

Definition 2 *A pattern $w =< x_1 ... x_{l-1} x_l >$ is frequent and relevant iff $P(q_0, w)$ is higher than a support threshold σ, and the proportion constraint on $P(q_0, w)$ and the dependence constraint between w and $w' =< x_1 ... x_{l-1} >$ are satisfied.*

4.2 The ACSM Algorithm

Combining all the concepts we presented so far, we propose a new constraint-based sequence mining algorithm. It aims at discovering from a PDFA all the fre-

Input: A PDFA A = (Q, Σ, q, q_0, π, π_F), a support threshold σ, two risks α_1 and α_2
Output: a set of relevant frequent patterns G

```
1  begin
2      G_1 ← ∅ ;
3      foreach l ∈ Σ do
4          if P(q_0, l) ≥ σ then
5              if PROP_TEST (P(q_0, l), α_1) is satisfied then
6                  G_1 ← G_1 ∪ {l} ;
7
8
9      G ← G_1 ;
10     n ← 1 ;
11     while G_n ≠ ∅ do
12         G_{n+1} ← ∅ ;
13         foreach w =< x_1...x_n >∈ G_n do
14             foreach w' =< x'_1...x'_n >∈ G_n do
15                 if MERGE (w, w') then
16                     v ← < w > . < x'_n > ;
17                     if P(q_0, v) ≥ σ then
18                         if PROP_TEST (P(q_0, v), α_1) then
19                             if CHI2_TEST (V_in, V_out, α_2) then
20                                 G_{n+1} ← G_{n+1} ∪ {v} ;
21
22
23
24
25
26
27         G ← G ∪ G_{n+1} ;
28         n ← n + 1 ;
29     return G ;
30 end
```

Algorithm 1: Pseudo-code of ACSM

quent and relevant patterns, according to a support threshold σ and two statistical risks α_1 and α_2. The pseudo-code of our ACSM algorithm (Automata-based Constraint-based Sequence Mining) is presented in Algorithm 1. From lines 2 to 9, it initializes a set of relevant frequent patterns composed of only one symbol. Since no pattern has been extracted yet, only the support test (line 4) and PROP_TEST (line 5) are run. The paths of the PDFA that do not satisfy these two tests will not be studied anymore, that allows us to prune the search space. The second part of ACSM tests additional symbols to search for larger frequent and relevant patterns. Three conditions must be satisfied: the support test (lines 17), PROP_TEST (line 18) and CHI2_TEST (line 19). The boolean function MERGE(w, w') (line 15) returns *true* if the $n-1$ last symbols of w are identical to the $n-1$ first ones of w'. This ensures that all subsequences of the resulting pattern $v =< w > . < x_n >$ (line 16) are yet frequent and relevant.

5 Car Flow Modeling

5.1 Traffic Miner

In order to bring to the fore the interest of our approach, we design a road traffic simulator, called TRAFFICMINER (see Fig.4). We simulated a road traffic on a map captured from mappy^{TM}'s web site. We implemented a graphical interface which allows us to model the map in a graph form: one-way and two-way roads as transitions of the PDFA, entry gates, exit gates and crossroads respectively as initial, final and other states. On each street, we put a counter to get the number of cars going through it. When the map is modeled, we can simulate the traffic by generating a random flow of cars. At any moment, we can stop the flow and get a PDFA $A =< Q, \Sigma, q, \pi, \pi_I, \pi_F >$ where: Q is the set of crossroads, and enter and exit gates; Σ is the set of street names; $q : Q \times \Sigma \rightarrow Q$ defines a transition, *i.e.* a street between two crossroads; $\pi : Q \times \Sigma \rightarrow [0, 1]$ associates a probability to each pair (S,z), *i.e.* the probability to leave the crossroad S taking the street z (as defined in Eq.1); $\pi_F : Q \rightarrow [0, 1]$ associates to each final state (*i.e.* the exit gates) a non-null probability $\pi_F(S)$ to leave the map taking the gate S; $\pi_I : Q \rightarrow [0, 1]$ associates to each initial state (*i.e.* to entry gates) a non-null probability $\pi_I(S)$. According to Def.1, a PDFA must have only one initial state to be deterministic. In our case, despite the fact that we have several initial states (entry gates), the determinism is not challenged because there does not exist two paths, starting from two initial states, that use the same transition.
To simulate the road traffic, a multinomial distribution is applied on the entry gates, and others are used on each crossroad to simulate the routes.

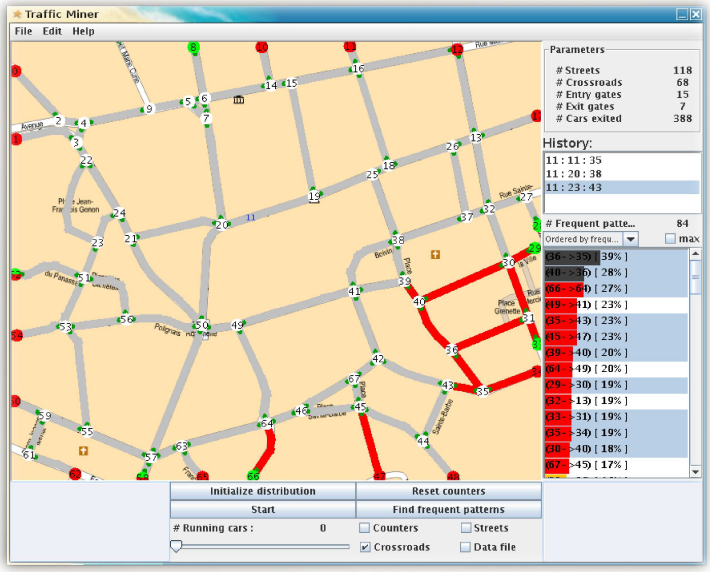

Figure 4. TrafficMiner

5.2 Interest of a Car Flow Modeling

From this PDFA, we can run ACSM to extract patterns that may be very interesting in many domains. First, it may

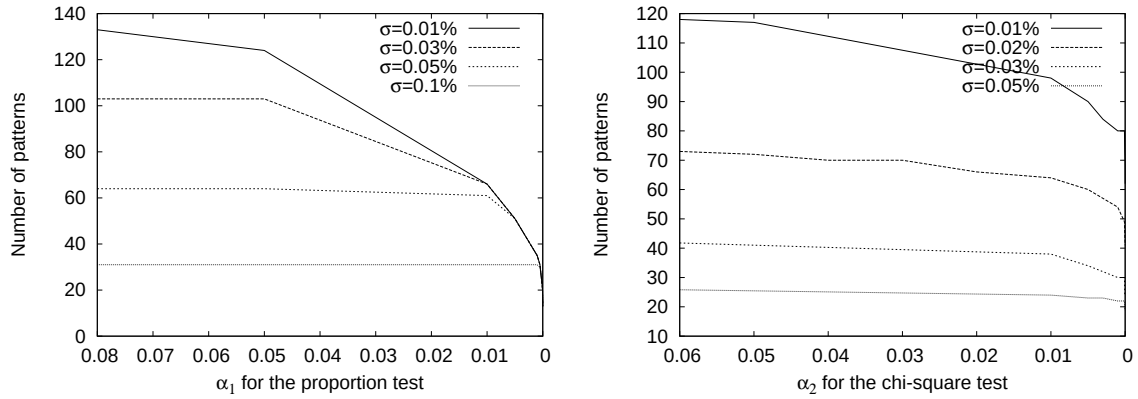

Figure 5. (a) and (b): Effect of constraints on the number of patterns.

be efficiently used in road traffic regulation. By finding frequent paths taken by the same cars, one could locate places in the map which would deserve some developments (traffic circles, traffic lights, etc.) to make the traffic more fluid. For example, in Fig.4, we can locate on the right a place where the traffic is very heavy. Second, it could be used to simulate a new traffic organization (modification of street directions, creation of new one-way streets, etc.) to avoid traffic jams. Finally, it could be useful in campaign advertising. Note again that a frequent and relevant pattern $w =< x_1 x_2 >$, extracted with our model, would express, *without any information about the individual trail of the drivers*, that the majority of cars taking the street x_1 will probably also take later the (possibly not adjacent) street x_2. This is the case of the pattern composed of the two black streets at the bottom in Fig.4. Our system is able to ensure that those who take the first street will also take the second one.

This kind of information could help in many domains. First, an advertising agency could use such techniques to find the best strategic position for billboards: either repeat the same hoarding to increase the effect of the advertisement or put a different one. Moreover, such information could also be useful for determining an optimal traffic regulation or simulating a new traffic organization.

5.3 Experimental Results

The main contribution of this paper is to show that one can achieve with a PDFA a constrained sequence mining task while preserving the privacy of the data. The experiments we carried out in Section 2 have proven the ability of the PDFA to succeed without constraints.

Effect of the Constraints. Let us now evaluate the individual effect of our constraints on the number of extracted patterns. We run then our simulator. Fixing $\theta = 0$, the two charts of Fig.5 show the effects of the relevance constraints. We tested the influence of PROP_TEST without incorporating CHI2_TEST (first chart, fixing $\alpha_2 = 100\%$) and reciprocally (second chart, fixing $\alpha_1 = 100\%$). Of course, we can note that the stronger one of these constraints is, the more the number of patterns decreases. Moreover, we can note that the more σ increases, the more the relevance constraints become obviously useless.

Experimental Comparison with SPAM**.** Since we do not use the original data, a comparison with other sequence mining algorithms seems more difficult. To overcome this drawback, we sampled sequences from the PDFA and tried to find back the same patterns with SPAM[2][3]. The experimental setup was the following. We simulated a flow of cars in our map, and we run ACSM to extract the patterns (here, without constraints to allow us the comparison with SPAM). We measured the time complexity (called ACSM time). From the PDFA, we sampled many sets of sequences (from 100 to 35,000 sequences). From each sample, we run SPAM to extract a second set of frequent patterns. Both of the algorithms were run with $\sigma = 10\%$. We computed also the time complexity of SPAM by taking into account the sampling time and the mining time (SPAM time). The chart of Fig.6 describes the behavior of the two methods.

While ACSM has a constant time complexity, the one of SPAM increases a lot in function of the sequence set size. We added on this chart a curve (Cover) corresponding to the proportion of frequent patterns extracted by SPAM which

[2]http://himalaya-tools.sourceforge.net/Spam/

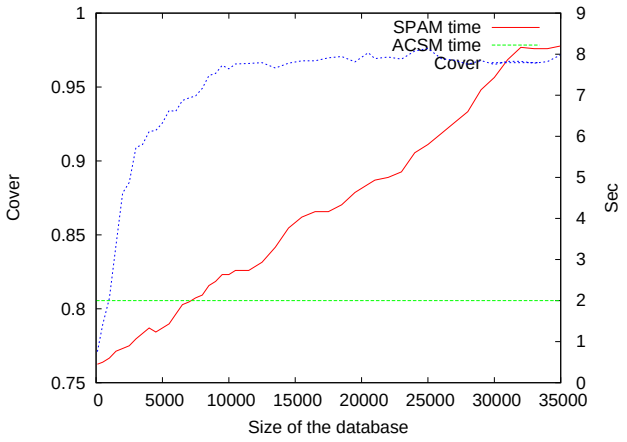

Figure 6. Comparison SPAM vs ACSM.

have also been extracted by ACSM. We can observe that SPAM needs a large number of sequences to approach the results of ACSM. We can note that once the size of the set is sufficiently large, about 20,000 examples, more than 97% of all the frequent patterns extracted from the PDFA are covered by the those founded by SPAM. But in this case, the time cost of SPAM is higher than the one of ACSM.

6 Conclusion

In this paper we have shown that, in situations that can be modeled as flow control problems, a sequence mining task may preserve privacy thanks to an underlying probabilistic automaton. We have proposed a new constrained sequence mining algorithm based on that data structure and shown this approach is more efficient than sampling a database from the PDFA and using a classical sequence mining algorithm. Our algorithm has been implemented in a prototype we have used to visually show the frequent routes of towns without any private information from its drivers. In the future, we want to use ACSM on two flow control problems: Web usage mining by using a PDFA to model the structure of a site with the flow of visits, and social network modeling by using a PDFA to model the flow of emails between people. In order to be as efficient as possible, it will be interesting to integrate other constraints such as mingap, maxgap, etc.

References

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proc. of the Inter. Conf. on Data Engineering*, pages 3–14. IEEE, 1995.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Proc. of the Int. Conf. on Management of Data*, pages 439–450. ACM, 2000.

[3] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu. Sequential pattern mining using a bitmap representation. In *Proc. of the Int. Conf. on Knowledge Discovery and Data Mining*, pages 429–435. ACM, 2002.

[4] R. Bayardo and R. Agrawal. Data privacy through optimal k-anonymization. In *Proc. of the Int. Conf. on Data Engineering*, pages 217–228. IEEE, 2005.

[5] J. Borges and M. Levene. Mining association rules in hypertext databases. In *Knowledge Discovery and Data Mining*, pages 149–153, 1998.

[6] J. Borges and M. Levene. Generating dynamic higher-order markov models in web usage mining. In *Proc. of the European Conf. on Principles and Practice of Knowledge Discovery in Databases*, pages 34–45. LNCS 3721, 2005.

[7] A. V. Evfimievski, R. Srikant, R. Agrawal, and J. Gehrke. Privacy preserving mining of association rules. *Information Systems*, 29(4):343–364, 2004.

[8] M. Garofalakis, R. Rastogi, and K. Shim. Mining sequential patterns with regular expression constraints. *IEEE Trans. on Know. & Data Eng.*, 14(3):530–552, 2002.

[9] J. Han, R. B. Altman, V. Kumar, H. Mannila, and D. Pregibon. Emerging scientific applications in data mining. *CACM*, 45(8):54–58, 2002.

[10] P. Hingston. Using finite state automata for sequence mining. In *Proc. of the Australasian Conf. on Computer science*, pages 105–110, 2002.

[11] F. Jacquenet, M. Sebban, and G. Valétudie. Mining decision rules from deterministic finite automata. In *16th IEEE Int. Conf. on Tools with Artificial Intelligence (ICTAI 2004)*, pages 362–367, 2004.

[12] M. Klemettinen, H. Mannila, and H. Toivonen. Interactive exploration of interesting findings in the telecommunication network alarm sequence analyzer. *Information & Software Technology*, 41(9):557–567, 1999.

[13] R. Kosala and H. Blockeel. Web mining research: A survey. *SIGKDD Explorations*, 2(1):1–15, 2000.

[14] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.*, 1(3):259–289, 1997.

[15] E. M. Newton, L. Sweeney, and B. Malin. Preserving privacy by de-identifying face images. *IEEE Trans. Knowl. Data Eng*, 17(2):232–243, 2005.

[16] J. Pei, J. Han, and W. Wang. Mining sequential patterns with constraints in large databases. In *Proc. of CIKM 2002*, pages 18–25. ACM Press, 2002.

[17] M. Spiliopoulou and C. Pohle. Data mining for measuring and improving the success of web sites. *Data Mining and Knowledge Discovery*, 5(1/2):85–114, 2001.

[18] L. Sweeney. k-anonymity: a model for protecting privacy. *Int. Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5):557–570, 2002.

[19] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *SIGMOD Record*, 33(1):50–57, 2004.

[20] M. J. Zaki. Sequence mining in categorical domains: incorporating constraints. In *Proc. of CIKM 2000*, pages 422–429. ACM, 2000.